Corporate Smalltalk Consulting Ltd

Help

Home

Oct 6th 1999.

POSTSCRIPT: A Page Description Language

By John M McIntosh

What's all this fuss about POSTSCRIPT, and why does XXXX recommend purchasing those expensive POSTSCRIPT printers? Lets take a look. First a quick overview, then with a more detailed description.

History.

POSTSCRIPT is the creation of Adobe Systems, which was founded by John Warnock & Charles Geschke. It was first marketed on the Apple LaserWriter in early 1985, and successfully competed against HP, XEROX, and IBM in the so called PDL *wars* of 1986/87 to become the graphics standard on most, if not all laser printers.

What is a PDL?

Originally printers viewed text as text, and graphics as graphics. A page was created from a conglomeration of separate entities. You supplied *raw* text like, the word *Hello*, combined with *Escape* codes and possibly embedded graphic pictures, to create page images. Of course you could choose just to send *raw* text to a printer and usually expect to get some sort of printed output.

On the other hand, a Page Description Language defines a true computer programming language which is specifically designed to create and modify both text and graphic images, with full equality on a page at any resolution and in any color or density! Instead of sending *raw* text to the printer, a POSTSCRIPT program is created and sent to the printer. A specialized computer within the printer running a POSTSCRIPT interpreter program runs the supplied program to create the requested page image. The printer's drawing engine (the machinery that puts the black toner on the paper), then takes the image and draws it on the page. This is a quite different concept from formatting the page image on the host computer. It alleviates computer applications from worrying about creating page images since the image creation is actually done by the printer.

The fact that POSTSCRIPT is a true programming language, supports color, and supports resolution from 300 DPI to 2540 DPI on such diverse hardware platforms as desktop printers up to typesetting quality printing presses, makes it a better and longer lasting solution.

FONTS

In the past most laser printers stored fonts, also known as typefaces, in a *Bit Image* format. A *Bit image* of a character can be thought of as a picture made up of dozens if not hundreds of tiny dots;

one is required for each character and for its various sizes. Because printer have limited permanent storage they could only support a small number of standard typefaces and sizes when using this method.

POSTSCRIPT on the other hand supports *Scalable* fonts, and if you remember from my earlier article on fonts, ADOBE wisely licensed the rights to various common typefaces from their respective owners. Now a *Scalable* font is actually a geometric description of a typeface's characters, and because it is a mathematical formula you can create any size character and perform transformations like text rotation. ADOBE fonts also have extensions called "*Hints*", which I will discuss later. Adobe's licensing of typeset standard fonts along with the þexibility of text sizing and the placement abilities of the POSTSCRIPT language, combined to make it a force which could not be ignored in the computer and typesetting industry, and thus made it the standard it is today!

COSTS

Ironically the features that make POSTSCRIPT such an attractive solution, also conspire against it. For example:

1. Most people don't want to create POSTSCRIPT programs, so software vendors need to provide the "hidden" support in their applications. On the Macintosh, Apple provided a POSTSCRIPT print driver, which allows Mac applications to transparently work with POSTSCRIPT printers. However on the PC-DOS side, each application requires their own POSTSCRIPT drivers. Even today some major PC applications still lack such support.

2. Since the printer now runs a computer program, the time and energy to produce each printed page is increased. Running the Interpreter requires a fairly fast computer, and at least 2048K of fast expensive memory. So within each POSTSCRIPT printer there is actually a high speed computer which is similar in computer power as the 2 Megabyte Macintosh SE, *sans* hard drive and screen. Until the Macintosh II was released, the joke around Apple was that the largest and fastest computer they sold was actually the Apple LaserWriter Plus printer.

Once you understand that you are licensing a large number of typefaces, buying a specialized fast computer with lots of memory, and purchasing the POSTSCRIPT software, it is easily realized why POSTSCRIPT capable printers are much more expensive than the HP Laserjet printer, which only contains the basic print engine.

Now let's review the above in some more detail:

History

John Warnock originally worked at XEROX PARC and was responsible for the development of INTERPRESS, which is XEROX's proprietary page description language. POSTSCRIPT was his better idea but XEROX choose not to pursue it. Warnock however did, and left to form Adobe. Charles Geschke provided the initial programming support. He came from the University of Utah graphic pioneer's group known as the Utah Mafia. Other famous people like Ivan Sutherland (Sketchpad) and Alan Kay (The Dynabook) also came from this group. Adobe is currently pursuing the goal of making *Display* POSTSCRIPT the standard for doing computer screen graphics on the next generation of computers, along with making POSTSCRIPT printers run much faster.

Escape codes

The dominant standards at this technology level are the EPSON escape codes for dot matrix printers, and the HP Laserjet codes for laser printers. For example, to print the text "Hello" you can send the following sequence of characters to a HP LaserJet. (*Esc* stands for the ASCII decimal code 27).

```
escE Reset the printer to its startup state
esc(8U Pick the Roman-8 font (Courier 12 point 10 pitch)
Hello Say Hello
esc&l0H Eject the printed page
An equivalent POSTSCRIPT program to say "Hello" is:
/in { 72 mul } def %define a POSTSCRIPT subroutine to translate
%inches into points (72 points to the inch)
gsave %Save the state of the graphics printer
/Courier findfont %Find the Courier font
12 scalefont setfont %Scale it to 12 points, select it
0 0 translate %Set the position at lower left of the page
.5 in 10.0 in moveto %Now come up 10 inches and over 1/2 inch
(Hello) show %Draw the characters Hello
grestore %Restore the old state of the printer
showpage %Eject the page
```

The POSTSCRIPT Language can be viewed in two parts, as a language, and as an imaging model. As an imaging model it defines how graphical objects are defined, filled, rotated, transformed, and placed on a page. It treats all objects including text in the same manner. As a language it has a vocabulary, a set of operators and syntactical rules. It is similar in syntax and function as the FORTH programming language, and as demonstrated, it has the ability to allow you to define and later use subroutines. The above example has a subroutine which translates inches into points.

TEXT & FONTS

As mentioned previously, a "*Hint*" is an ADOBE proprietary method which modifies a typeface's outline when working at smaller sizes to make each character more pleasing to the eye. Because of this, Adobe fonts look better in smaller sizes than other third party *Scalable* fonts which do not have "*Hints*".

Since *Scalable* fonts are geometric descriptions rather than *Bit Images* , in theory it is possible to create characters as small as 1/72 of an inch, and as large as Rhode Island. They can also be rotated to any angle, distorted, and even filled with patterns and still retain their original design integrity. In fact, many of the object/text transformation abilities such as rotation, stretching, and the slanting abilities contained in the Macintosh version of SuperPaint, are actually based on the power of POSTSCRIPT.

Adobe has actually licensed hundreds of typefaces from different owners. Fortunately, most printer vendors opted to purchase a standard set of 35 odd type faces from them. This *defacto* standard allows XXXX to purchase printers from IBM, QMS or Apple and still maintain a consistency across printers.

Costs:

Most vendors choose to integrate the POSTSCRIPT engine within the printers so they could easily

share the printer between various computers in a local area network. IBM did attempt to place the procXXXXr on a PC CARD, but the printer then was not shareable and did not sell very well. Their second attempt followed the industry acceptable standards. Most vendors use the Motorola 68000 procXXXXr because of its 16 Megabyte memory address space and fast processing speed. They use the 68020 chip, which is the standard for engineering workstations, on their high end printers. Some vendors are now experimenting with RISC based graphics procXXXXrs to provide the required computing horsepower.

Since POSTSCRIPT is a publicly defined computer language, we are now seeing third party POSTSCRIPT Interpreter CLONES being introduced in the market place along with cloned typefaces. These third party solutions should drive the cost of the true POSTSCRIPT printers down.

Examples:

Since the Macintosh version of MS-WORD supports native POSTSCRIPT, you too can play with POSTSCRIPT. Copy the follow programs exactly as shown, select all the lines in WORD, and hit the apple-shift-S keys. This action will allow you to input a STYLE name in the lower left of the window. Type PostScript as the STYLE name and hit the Return key. At this point WORD should change the text to the bold typeface. If in doubt, consult the MS-WORD user manual for clarification.

This selected style tells WORD to pass the affected text as POSTSCRIPT commands directly to the printer, versus printing it on the page. More advanced users can use this feature to enhance their output with features unattainable in WORD.

Words of warning:

Syntax errors in the POSTSCRIPT program will cause the printer to abort the program without any indication of the error. In addition, some complex POSTSCRIPT programs can run a LONG time, hours in fact. It is not popular to monopolize a shared printer for an hour or so. Remember it is a programming language and infinite loops due to programming errors can result.

Programs:

The first example draws circles of random sizes from 0 to 100 points at random locations on the page. 60% of the time it will fill the circle with a gray shade of random density. It should only take a few seconds to run.

```
% Circle drawing program
/rot { 3 -1 roll } def
/genRand { rand 100 mod 100 div } def % Calculate a random number 0-1
/randPoint { 612 genRand mul 792 genRand mul } def % Calculate place on the page
/circle { rot
0 360 arc
genRand 0.4 gt { genRand setgray fill 0 setgray } if
stroke
} def
/placeCircle { 100 genRand mul randPoint circle } def
/bubble { 0 0 moveto
70 { placeCircle } repeat % Draw 70 circles
} def
gsave
```

```
0 0 moveto
bubble
grestore
```

Note that the % character signifies a following comment or comment line to POSTSCRIPT.

The second example takes a specific character string and draws it five times on the page at a decreasing density and angle. It should only take 10 seconds or so to run.

```
/rotangle 90 def % define the start angle
/graylevel .9 def % define the starting gray level 0.00 to 1.00
/rotoffset 22.5 def % define the degrees to decrement by
/grayoff 15 def % define the percent to decrement graylevel by.
/fontsize 90 def % define the size of the text
/howmany 5 def % define the number of times to rotate the text
/thetext (McIntosh) def % define the text to use
/in { 72 mul } def % define a conversion subroutine for inchs to points
/rotationloop { % define the rotation subroutine
gsave
0 0 moveto %Set text location
rotangle rotate %Set rotation value
graylevel setgray %Set the gray level
thetext show %Draw the text
/rotangle rotangle rotoffset sub def % subtract rotooffset from rotangle
/graylevel graylevel grayoff sub def % substract grayoff from graylevel
grestore
} def % End of subroutine
gsave % save the graphic state
/Times-BoldItalic findfont % choose TIMES Bold italics as the font
fontsize scalefont setfont % set the font size and make it the default
3.0 in 5.5 in translate % place the text at 3.0 over and 5.5 inches up
howmany { rotationloop } repeat % run the rotationloop subroutine 5 times
grestore
```

Both these examples are fairly simple. More complex and longer running POSTSCRIPT programs created the following images.

[images here]

If you want to learn more about POSTSCRIPT, Adobe has a newsletter you can subscribe to called "***Colophon***". They have also written two text books on POSTSCRIPT.

PostScript Language Reference Manual

PostScript Language Tutorial and CookBook

Both books are published by Addison-Wesley and are available from the finer book stores in town.

Definition for: **page description language**

SEARCH

Look Up

(PDL).A language that describes to the printer how to print a page and how to form the type and graphic elements that appear on it.Rather than download a font to the printer, the page description language gives the printer instructions for how to draw each character from outlines; graphics are produced in a similar way.Examples are Adobe PostScript and Xerox Interpress.

Definition: page description language

Search dictionary for page description language

Source: *The Free On-line Dictionary of Computing (09 FEB 02)*

Page Description Language

(PDL) A language such as Adobe Systems, Inc.'s PostScript or Xerox's Interpress which allows the appearance of a printed page to be described in a high-level, device-independent way. Printing then becomes a two-stage process: an application program produces a description in the language, which is then interpreted by a specific output device. Such a language can therefore serve as an interchange standard for transmission and storage of printable documents.

(1995-02-20)